เรื่องน่าอ่าน
ดร.ปัญญา เปรมปรีดิ์

การออกแบบระบบที่ใช้ประโยชน์จากเครือข่ายอย่างเต็มที่

แนะนำวิธีออกแบบระบบที่จะช่วยให้ท่านใช้ประโยชน์จากเครือข่ายต่าง ๆ ได้อย่างเต็มที่

ได้อ่านบทความเรื่อง "การประยุกต์งานบนระบบเครือข่ายคอมพิวเตอร์" ของคุณพันธ์ปิติ (วารสารไมโครคอมพิวเตอร์ ฉบับที่ 97 หน้า 202) ที่ได้ทิ้งท้ายไว้ว่า "หวังว่าผู้ที่อยากมีเน็ตเวอร์กหรือมีเน็ตเวอร์กอยู่แล้วคงจะมีระบบงานที่ไม่ใช่มีไว้คุยกับชาวบ้าน หรือเป็นของเล่นสนองความอยากรู้เท่านั้น ผลสำเร็จของเน็ตเวอร์กคือ ใช้งานได้ ไม่ใช่ว่าลองต่อแล้วติด ฯลฯ" ผมได้ลองถามคนที่มียูนิกซ์เน็ตเวอร์ก แล้วพบว่าเขาหาแอพพลิเคชันที่ต้องใช้ซีพียู-ทู-ซีพียู-คอมมูนิเคชันไม่ได้ ส่วนใหญ่ก็เล่นกันแค่อิเล็กทรอนิกส์เมล์ภายในห้องเดียวกัน ซึ่งผมก็ไม่รู้ว่าจะทำไปทำไม การติดต่อไปยังสมาชิกในต่างประเทศนั้นก็เป็นอีกเรื่องหนึ่ง ซึ่งผมไม่แน่ใจว่ามันคุ้มกับค่าโทรศัพท์ข้ามประเทศหรือเปล่า (เรื่องนี้อาจมีข้อยกเว้นสำหรับอาจารย์บางท่านซึ่งต้องติดต่อกับ ISO เพื่องานของ สมอ.) นอกจากเรื่องเหล่านี้แล้ว ผมก็เห็นแต่งานประเภทที่ใช้ซีพียูเดียวก็เพียงพอแล้ว แต่เขาก็พยายามเอามันมาเชื่อมโยงกัน

แต่อย่างไรก็ตาม มันก็น่าเห็นใจเพราะงานที่เข้าลักษณะที่ต้องใช้หลาย ๆ ซีพียูช่วยกันทำงานนั้นหายาก แม้แต่พีซี-แลนที่ผมมีอยู่ถึง 20-30 วงนั้น ส่วนใหญ่ก็ทำงานแบบ Centralized (ถ้าตีความตามแบบของคุณพันธ์ปิติ) มันจึงมีคำถามขึ้นมาเหมือนกันว่า ที่เราใช้แลนกันอยู่ทุกวันนี้มันเป็นการเหมาะสมหรือไม่ และถ้าจะให้มันได้ใช้ประโยชน์จากเน็ตเวอร์กจริง ๆ เราต้องออกแบบระบบกันอย่างไร

ให้ลูกข่ายทำงานร่วมกับเซอร์ฟเวอร์

ผมจะเริ่มจากการวิเคราะห์การใช้พีซี-แลนของผมเสียก่อนผมจะขอยกตัวอย่างให้ดูสัก 3 เรื่อง เรื่องแรกเป็นระบบบริการซื้อขายพันธบัตร เรื่องนี้เราใส่ไฟล์เซอร์ฟเวอร์เข้าไป 2 ตัว มันเกาะอยู่บนสายแลนอันเดียวกัน แล้วเราก็มีเครื่องพีซีเป็นลูกข่ายอีกประมาณ 30 ตัว ลูกข่ายเหล่านี้ส่วนใหญ่จะใช้เพื่อบริการซื้อขาย โอน จำนอง แบ่ง

แยก จ่ายดอกเบี้ย อีกจำนวนหนึ่งใช้เพื่อเรียกดูข้อมูลและเพื่อสั่งให้ทำรายงาน ลูกข่ายประเภทแรกกับประเภทเรียกดูข้อมูลนั้นจะทำงานร่วมกับเซอร์ฟเวอร์ตัวแรก ส่วนลูกข่ายประเภทสั่งทำรายงานนั้นใช้เซอร์ฟเวอร์ตัวที่ 2 แต่จริง ๆ แล้วมันทำงานทดแทนกันได้หมด เซอร์ฟเวอร์ตัวแรกนั้นจะบรรจุมาสเตอร์ไฟล์ตัวปัจจุบัน เมื่อตกเย็นก็จะก๊อบปี้ไปลงเซอร์ฟเวอร์ตัวที่ 2 การสั่งทำรายงานก็จะได้ไม่ไปก่อกวนเซอร์ฟเวอร์ตัวแรก แล้วมันยังเป็นแบคอัพของตัวแรกด้วย

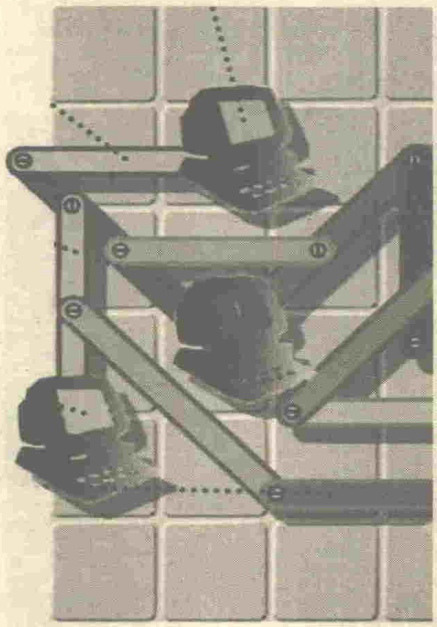
เมื่อดูตามนี้ก็จะเห็นว่าจริง ๆ แล้วเราสามารถแยกออกเป็นแลน 2 ชุดได้ ชุดหนึ่งใช้เพื่อบริการออนไลน์ที่หน้าเคาน์เตอร์ คือคนมาขอซื้อ ขาย โอน จำนอง แบ่งแยกรับดอกเบี้ย ก็จะบริการได้เลย ส่วนอีกชุดหนึ่งนั้นคือ ตัวแบคอัพและทำรายงาน ซึ่งเมื่อมองดูแต่ละชุดก็จะเป็นงานแบบ Centralized คือมีแฟ้มข้อมูลอยู่ที่ไฟล์เซอร์ฟเวอร์แล้วลูกข่ายก็มาเรียกดูและอัพเดตกันที่นี่ ดูอย่างนี้แล้วก็จะดูเสมือนว่าไม่ได้ใช้ประโยชน์จากการเชื่อมโยงเป็นเครือข่ายเลย แต่ผมก็ยังอยากจะให้ท่านลองพิจารณาข้อคิดเห็นของผม 3 ข้อ คือ

1. ทดแทนเครื่องเมนเฟรม

ระบบนี้ลงทุนไปประมาณ 4 ล้านบาท มันมาทดแทนเครื่องเมนเฟรมยี่ห้อ SPERRY ซึ่งมีราคาค่าเช่าปีละ 16 ล้านบาท อย่างนี้จะเรียกว่าเป็นการใช้ประโยชน์ได้ไหม ผมคิดว่าได้ แต่ไม่รู้ว่าท่านผู้อ่านคิดอย่างไร

2. ใช้ซีพียูหลายตัวพร้อมกัน

ระบบพีซี-แลนแบบที่กล่าวมาข้างต้นนี้ จริง ๆ แล้วมีการใช้ประโยชน์ของ



ซีพียูหลายตัวพร้อม ๆ กัน คือโปรแกรมที่วิ่งอยู่จริง ๆ นั้นมี 2 ส่วน ส่วนหนึ่งอยู่ในซีพียูของเซอร์ฟเวอร์ อีกส่วนหนึ่งอยู่ในซีพียูของลูกข่าย มันทำงานผสานกัน ตัวลูกข่ายนั้นเป็น Client ส่วนในเซอร์ฟเวอร์นั้นเป็น Server แต่หน้าที่ของเซอร์ฟเวอร์มันดูจะง่าย คือทำหน้าที่อ่านและบันทึกข้อมูลลงจานแม่เหล็กให้เราเท่านั้น โปรแกรมส่วนนี้ผู้ใช้ไม่ต้องเขียน เราซื้อ B-Trieve มาใช้งาน มันสำเร็จรูปมาจากโรงงานเลย อย่างนี้จะเรียกว่าเป็นการใช้ประโยชน์ของเครือข่ายได้ไหม ผมคิดว่าใช่ แต่มันเป็นการใช้ประโยชน์ในรูปแบบที่พื้นฐานมาก ท่านคิดอย่างผมไหม

3. ลูกข่ายทำงานทดแทนกันได้

การที่เราเอาเซอร์ฟเวอร์และพีซีทั้งหมดมาเกาะบนสายแลนอันเดียวกันนั้นจะช่วยให้ลูกข่ายทำงานทดแทนกันได้ และถ้าเซอร์ฟเวอร์ตัวหนึ่งตัวใดเสียไปเราก็จะใช้อีกตัวหนึ่งทำงานแทนได้ (ต้องใช้เวลาในการโหลดไฟล์กันพอสมควร) นอกจากนี้แล้วเวลาเราสั่งก๊อบปี้ไฟล์จากเซอร์ฟเวอร์ตัวหนึ่งไปยังอีกตัวหนึ่งมันก็เป็นการใช้ประโยชน์ของเน็ตเวอร์ก เรื่องนี้ท่านเห็นด้วยกับผมไหม

แบ่งระบบย่อยหลายระบบ

ยังไม่ต้องตอบคำถามของผมหรอก และจริง ๆ แล้วก็คงไม่ต้องตอบเลย เพราะผมเขียนมาเพื่อให้เราคิดกันเล่น ๆ ผมจะขอผ่านถึงตัวอย่างที่ 2 ตัวอย่างที่ 2 เป็นเรื่องของระบบการพนักงาน ระบบนี้เราใส่ไฟล์เซอร์ฟเวอร์เข้าไป 1 ตัว ใส่ดาต้าเบสเซอร์ฟเวอร์เข้าไปอีก 1 ตัว นอกนั้นเป็นลูกข่าย ซึ่งมีอยู่ประมาณ 18 ตัว ระบบการพนักงานนี้จริง ๆ แล้วแบ่งเป็นระบบย่อยหลายระบบ แต่ในแง่ของการทำดาต้าโปรเซสซิงแล้วคล้ายกันหมด คือในตอนกลางวันเราจะใช้ลูกข่ายเพื่ออัพเดตและเรียกดูมาสเตอร์ไฟล์ที่อยู่ในดาต้าเบสเซอร์ฟเวอร์ มันวิ่งอยู่ภายใต้ Oracle คนทำงานจริง ๆ ก็ไม่เต็ม 18 จอหรอก เพราะมีการแก้ไขเปลี่ยนแปลงข้อมูลไม่มาก แต่ที่เราต้องใส่จอลูกข่ายเข้าไปมาก ๆ ก็เพราะต้องกระจายไปให้ทั่วทุกหน่วยที่เกี่ยวข้อง พวกเขาอาจมาเรียกดูข้อมูลเมื่อไหร่ก็ได้ และจะใช้โปรแกรมที่เราทำไว้ให้ หรือจะสร้างคำถามตามแบบ SQL ก็ได้ แต่เราจะไม่แนะนำให้ทำรายงานอะไรที่กินกำลังเครื่องมาก ๆ เช่น ค้นหาคนทุกคนที่มีเงินเดือนสูงกว่า 20,000 บาท เพราะงานแบบนี้ค้นหมดทุกเรคอร์ด แล้วลากเอาไปเก็บลงดิสค์ให้เต็มไปหมด รายงานประเภทนี้เราให้รอตอนเย็น ซึ่งจะใช้วิธี Extract ข้อมูลลงเป็น Sequential File แล้วโปรเซสในไฟล์เซอร์ฟเวอร์

ธรรมดา ด้วยวิธีนี้เราออกรายงานได้เร็วกว่า

คำถามของผมก็จะเหมือน ๆ กับในตัวอย่างแรก คือเราพัฒนางานนี้ขึ้นมาทดแทนระบบที่วิ่งแบบกึ่งออนไลน์กึ่งแบตช์โปรเซสซิงในเครื่องเมนเฟรม มันลดค่าใช้จ่ายลงได้ แล้วมันก็มีการใช้ประโยชน์ของซีพียู-ทู-ซีพียู-คอมมูนิเคชันอย่างนี้จะเรียกว่าเป็นการใช้ประโยชน์ของเน็ตเวอร์กหรือเปล่า ผมคิดว่าใช่ แต่มันอยู่ในรูปแบบที่ไม่หรูหราเท่าใดนัก

ใช้ Gate Way

สำหรับตัวอย่างที่ 3 นั้นผมคิดว่าหลาย ๆ ท่านก็คงจะมีประสบการณ์ คือในฝ่ายวิชาการของเรานั้นเขามีทั้งเมนเฟรมและพีซี-แลน เมนเฟรมเป็นไอบีเอ็ม 9370 ส่วนพีซี-แลนเป็นของโนเวลล์ บนเมนเฟรมนั้นเราใส่ Oracle ลงไป การทำ Querry จึงสะดวกดีแต่งานวิเคราะห์อื่น ๆ นั้นทำบนพีซีธรรมดา ๆ ได้ดีกว่า พวกเขาใช้ลูกข่ายในพีซีแลนเป็นตัวทำงานนี้ แต่ข้อมูลต้องมีการดึงมาจากเมนเฟรม และเมื่อนำมานวดเฟ้นจนดีแล้วก็จะอัพโหลดขึ้นไปบนเมนเฟรม ดังนั้นเราจึงมีความจำเป็นต้องเชื่อมโยงพีซี-แลนนี้เข้ากับเมนเฟรม เรื่องนี้เราใช้ Gate Way ธรรมดา ๆ คือ ยอมเสียสละพีซีตัวหนึ่งบนแลนแล้วต่อสายไปเข้าเมนเฟรม ด้วยวิธีนี้ทำให้ลูกข่ายทุกตัวสามารถติดต่อเป็นเทอร์มินัลของเมนเฟรมได้ การทำงานของระบบก็จะต้องใช้ซีพียูหลายตัวพร้อม ๆ กัน ซอฟต์แวร์ในเรื่องนี้เราซื้อจาก Third Party มันมีกี่ตัวผมก็ไม่ทราบ แต่ก็ทำงานได้ดีมากทีเดียว เทคนิคนี้มีใช้กันอยู่หลายแห่งในเมืองไทย คนอื่นอาจไม่ได้คิดว่ามันเป็นเน็ตเวอร์ก แต่ผมว่าใช่ และมันก็มีประโยชน์มากด้วย เพราะถ้าผมจะเปลี่ยนเทอร์มินัลทั้งหมดให้เป็นของไอบีเอ็มผมก็คงต้องอัพเกรดเครื่อง 9370 ไปเป็น ES/9000 ซึ่งต้องลงทุนอีก 50-60 ล้านบาท

ผมอยากจะสรุปว่าการใช้ประโยชน์ของเน็ตเวอร์กแบบพีซี-แลนนี้มันดูจะไม่หรูหราอะไร มันเป็นการประยุกต์ใช้เน็ตเวอร์กอย่างแบบตื้น ๆ แต่ก็มีปัญหาอยู่เสมอ ๆ คือคนไทยเรายังคิดไม่ค่อยออกว่าจะออกแบบระบบกันอย่างไร ในวันนี้ผมจึงอยากจะพูดแถมในเรื่องการออกแบบเสียหน่อย ซึ่งเป็นเรื่องหลักการที่สำคัญ โปรแกรมเมอร์ต้องเข้าใจจึงจะประยุกต์ใช้เน็ตเวอร์กได้

ติดอยู่กับเมนเฟรม

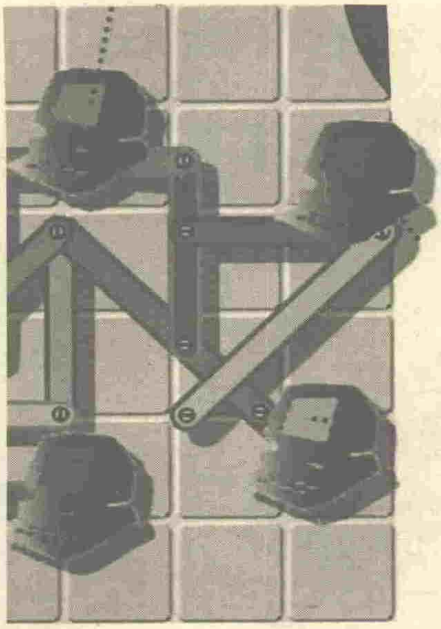
ปัญหาของโปรแกรมเมอร์เท่าที่ผมพบก็คือ เราผูกพันกับเมนเฟรมมานาน ถึงแม้จะมีพีซีเข้ามาในชีวิตของเราแต่เราก็ยังคิดแบบเมนเฟรม คิดแบบเมนเฟรมเป็นอย่างไรหรือครับ มันเป็นอย่างนี้ครับ ระบบเมนเฟรมนั้นมีโปรแกรมอยู่ในซีพียูเพียงแห่งเดียว ทุกสิ่งทุกอย่างจะถูกกำหนดให้กระทำอย่างต่อเนื่องโดยโปรแกรมนี้ เราต้องวางขั้นตอนทุกขั้นตอนให้อยู่ในโปรแกรมดังกล่าว เราสั่งให้รับข้อมูลจากเทอร์มินัล เราสั่งให้ตรวจเช็คข้อมูล เราสั่งให้อ่านจานแม่เหล็ก เราสั่งให้แก้ไขข้อมูล เราสั่งให้บันทึกกลับลงจานแม่เหล็ก เราส่งผลออกไปที่จอของเทอร์มินัลแล้วเราก็วนกลับไปเริ่มต้นรายการใหม่ ถ้ามีเหตุการณ์ผิดปกติตรงไหนเราก็เติมโปรแกรมให้จัดการกับปัญหานั้น โดยสรุปแล้ว Data Processing ถูกกระทำในโปรแกรมแห่งนี้แห่งเดียว งานจะทำไปทีละสเต็ปโดยไม่มีการขนานกัน ถ้าเราจะแยกเป็นโปรแกรมย่อยเราก็จะวิ่งกันคนละเวลา

นี่แหละครับคือคิดแบบเมนเฟรม ซึ่งจะใช้ไม่ได้กับระบบเน็ตเวอร์ก ระบบเน็ตเวอร์กแม้แต่ในระดับชั้นต่ำที่สุดคือพีซีแลนก็จะต้องคิดแยกงาน Data Processing ออกเป็นหลาย ๆ ส่วน แล้วแบ่งให้โปรแกรมในซีพียูต่าง ๆ ช่วยกันทำ ในการแยกงานออกไปนี้เขาก็มีหลักการซึ่งจะไม่เหมือนกับการแยกเป็นหลาย ๆ โปรแกรมในระบบเมนเฟรม หลักการสำคัญเป็นดังนี้ คือให้มองดูงานของลูกข่ายทุกคน แล้วแยกว่างานไหนเป็นการกระทำที่ต้องต่างคนต่างทำ และงานไหนเป็นงานที่เหมือน ๆ กัน ซึ่งจะทำได้โดยโปรแกรมใหญ่ในซีพียูอื่นเมื่อแยกได้แล้วก็จะเขียนโปรแกรมแยกกัน โปรแกรมเพื่องานประเภทแรกนั้นทำแบบ Sequential เหมือนในเมนเฟรม แต่งานประเภทที่ 2 นั้นจะต้องเขียนแบบหมุนเวียนไปบริการโดยไม่มีการหยุดรอ

ควรให้งานอยู่ในพีซีของลูกข่าย

เรื่องนี้ก็คงต้องมีการอธิบายและขยายความกัน วิธีแรกคือจะขอยกตัวอย่างให้ดู คือการรับข้อมูลจากคีย์บอร์ดและตรวจเช็คว่าเป็นตัวเลข เป็นตัวอักษร เป็นรหัสที่เป็นไปได้หรือไม่ มีค่าเช็คดิจิตถูกต้อง ฯลฯ เหล่านี้เป็นงานที่ต้องทำที่เทอร์มินัล มันต้องต่างคนต่างทำ ดังนั้นงานส่วนนี้ควรให้อยู่ในซีพียูของลูกข่าย แต่หลังจากตรวจดูเสร็จแล้วเราก็ต้องอ่านมาสเตอร์ไฟล์ อัพเดตมาสเตอร์ไฟล์ และแจ้งผลการอัพเดต งานในขั้นตอนที่ 2 นี้ทำเหมือน ๆ กัน และทำกับมาสเตอร์ไฟล์ตัวเดียวกัน ดังนั้นจึงควรแยกออกมาเป็นโปรแกรมที่วิ่งอยู่ในซีพียูที่คุมจานแม่เหล็ก การแจ้งผลนั้นก็ควรส่งเป็นแค่ข้อมูลมาตรฐานสั้น ๆ อันหนึ่ง แล้วไปปล่อยให้เป็นหน้าที่ของโปรแกรมในลูกข่ายที่จะไปขยายความเป็นภาพบนจอภาพของลูกข่ายอีกต่อหนึ่ง

สิ่งที่ต้องระวังในการเขียนโปรแกรมในซีพียูที่คุมจานแม่เหล็กก็คือ มันจะถูกเรียกใช้โดยโปรแกรมจากลูกข่ายหลายตัว มันจึงหยุดรอใครคนใดคนหนึ่งไม่ได้ เรื่องนี้ทำได้โดยต้องจัดให้มีคิวของคำสั่ง ลูกข่ายจะส่งคำสั่งเข้ามาอยู่ในคิว แล้วโปรแกรมตัวคุมจานแม่เหล็กจะหมุนเวียนไปดึงเอาคำสั่งในคิวมาทำงาน ทำเสร็จแล้วจึงหมุนวนกลับไปดึงจากคิวมาทำอีก เมื่อไหร่ที่คิวไม่มีอะไรเหลืออยู่เลยก็จะหยุดพักนอน 1-2 วินาที แล้วค่อยวนกลับไปเช็คอีก นี่เป็นวิธีหนึ่งที่จะทำให้โปรเซส 2 แห่ง (ในลูกข่ายกับในเซอร์ฟเวอร์) ไม่ถูกผูกติดกันด้วยเงื่อนเวลา ทั้ง 2 ต่างสามารถ



ทำงานเป็นสเต็ปไปได้เหมือนเดิม เพียงแต่ต้องรอคอยกันเป็นครั้งเป็นคราว

กระจายงานให้ซีพียูของลูกข่าย

มาถึงตรงนี้ผมคิดว่าเราคงพอจะเริ่มมองออกว่าจะแบ่งแยกงานอย่างไร แต่มันก็มีคำถามเกิดขึ้นได้ เช่น การตรวจเช็คข้อมูลที่เทอร์มินัลนั้นจริง ๆ แล้วก็มีลักษณะเหมือน ๆ กัน หรือคือต้องทำด้วยรูทีนเดียวกันทั้งนั้น ดังนั้นมันจึงอาจถูกย้ายมารวมอยู่ในโปรแกรมในเซอร์ฟเวอร์ก็ได้ เรื่องนี้เขามีกฎข้อที่ 2 คือ พยายามกระจายงานที่เหมือน ๆ กันเหล่านั้นออกไปอยู่ในซีพียูของลูกข่ายให้มากที่สุดเท่าที่จะมากได้ เหตุผลเรื่องนี้คือ ซีพียูของลูกข่ายนั้นมักจะว่างงาน ถ้าเราเอางานที่มีลักษณะเหมือน ๆ กันมาให้ตัวร่วมตรงกลางจนหมดเราก็จะทำงานได้ช้าลง ลูกข่ายจะต้องรอตัวศูนย์กลางมากขึ้น การแยกและกระจายงานนี้บางคนเขาบอกว่าเป็นศิลปะ มันไม่มีกฎเกณฑ์ที่ตายตัว ท่านต้องจัดให้มันสมดุลกันเอง แต่ผมชอบใช้แนวคิดตามกฎข้อ 2 นั้นมากกว่า

ก่อนจาก

จริง ๆ แล้วเรายังมีกฎเกณฑ์อื่น ๆ อีกหลายข้อ เช่น โปรเซสที่จะจับมารวมไว้ตรงกลางนั้นควรเป็นโปรเซสสั้น ๆ ง่าย ๆ และควรเขียนโปรแกรมให้มีประสิทธิภาพสูง นอกจากนี้แล้วเทคนิคการเรียกใช้โปรแกรมที่เป็นตัวร่วมอยู่ตรงกลางนี้ยังมีได้อีกหลายแบบ เช่น ทำเป็นรูทีนที่ถูกเรียกใช้แล้วคงค้างอยู่ในเมมโมรี (Terminate and Stay Resident, TSR) หรือวิ่งได้หลายก๊อบปี้ในเวลาเดียวกัน (Multi-task และ Multi-thread) สิ่งต่าง ๆ เหล่านี้เป็นเทคนิคที่จะช่วยให้ระบบเน็ตเวอร์กทำงานได้ดียิ่งขึ้น เทคนิคเหล่านี้จริง ๆ แล้วมีอยู่ในเมนเฟรม แต่มันถูกใช้ในการสร้างซิสเต็มส์ซอฟต์แวร์เท่านั้น คนอย่างเรา ๆ ไม่เคยได้สัมผัส ท่านต้องเป็นนักอ่าน นักคิด และนักทดลอง จึงจะมองเห็นช่องทางเหล่านี้

ซึ่งผมคิดว่าไม่เกินขีดความสามารถของพวกเรา